B. AMENDMENTS TO THE SPECIFICATION

Please replace the paragraph on page 4, lines 3-20 with the following:

A thread may voluntarily preempt by yielding processor resources and stalling temporarily. This may happen if a desired resource is unavailable or the thread needs to wait for a data signal. Typical preemptive services that may cause a thread to preempt include synchronization mechanisms like semaphores, mutexes, and the like. These services are used for inter-thread communication and coordinating activities in which multiple processes compete for the same resources. For instance, a semaphore, corresponding to a resource, is a value at a designated place in the operating system storage. Each thread can check and then change this value. Depending on the value found, the thread could use the resource or wait until the value becomes conducive to using the resource. Similarly, mutexes are program objects created so that multiple program threads can take turns sharing the same resource. Typically, when a program is started, it creates a mutex for a given resource at the beginning by requesting it from the system. The system returns a unique name or identification for it. Thereon, any thread needing the resource must use the mutex to lock the resource from other threads while using the resource. Another class of preemptive services is related to input-output and file access. Alternatively, a thread may preempt while waiting for a timer signal or a direct memory access (DMA) ~~DMA~~ transfer to complete. A thread may also be waiting for receiving access to a special-purpose processor or simply waiting for an interrupt.

Please replace the paragraph on page 8, lines 23-26 and page 9, lines 1-2 with the following:

The disclosed invention uses a new way of programming the threads. The threads are programmed using ~~uses~~ a series of multiple small tasks (called errands). The desired sequence of errands is given to the operating system for execution in the form of an itinerary. The programming methodology of the disclosed invention results in minimizing switching overheads as well as reducing the memory usage required for processing the threads.

Please replace the paragraph on page 11, lines 15-24 with the following:

Fig. 2B schematically illustrates the itinerary running service of the operating system. Itinerary running service 118 enables building and execution of threads in itinerary mode. In addition to preemption service 202, standard preemptive services 204 and non-preemptive services 206, it provides an itinerary building service 214. Itinerary building service 214 aids in the setting up of the itinerary. A computing job or an errand is written as a function and the function pointer is put in the itinerary through itinerary building service 214. If the errand requires any special data, such data is also saved in the itinerary data list through itinerary building service 214. In addition, itinerary building service 214 ~~112~~ facilitates passing of data from one errand to another by allocating space for variables on the itinerary data list.

Please replace the paragraph on page 12, lines 15-24 with the following:

In case the thread requests running an itinerary at step 306, the thread needs to be preempted and switched out of normal mode. The thread's context is stored in accordance with step 316. At step 318 the thread is preempted in the normal mode and enters itinerary mode. In the itinerary mode, the thread is executed through itinerary running service 118 ~~114~~ of the operating system. Once the thread enters itinerary mode, it continues to execute the errands on the itinerary until the entire itinerary is executed, in accordance with step 320. This step will be further elaborated upon in conjunction with Fig. 4. At step 322, upon completion of itinerary execution, the thread exits the itinerary mode. It calls the scheduler to schedule the next thread at the head of the ready queue.

Please replace the paragraph on page 16, lines 31-33 with the following:

The following example illustrates the manner in which a thread may be made to run completely within an itinerary. A standard thread which acts as ~~a~~ consumer for some thread, and as a producer for some other thread can be written as under.

Please replace the paragraph on page 18, lines 5-11 with the following:

The errand `computation` is a special errand written by a programmer and performs application specific computation. It is written as a normal function, the pointer

to which is stored on the itinerary function-list using a special function provided by itinerary building service 214 ~~112~~. In the above example, this function is represented as `itk_add_errand`. A similar function is used by the standard errands such as `errand_semaphore_wait`. The semaphore wait call can equivalently be written as follows.

Please add the following paragraph at page 26, line 9:

The present invention may also be embodied in computer program product for executing a multithreaded application. The computer program product includes a computer readable medium having program instructions comprising the multithreaded application code.